

BBBBBBBBBBBB		AAAAAAA		SSSSSSSSSS		RRRRRRRRRR		TTTTTTTTTTTT		LLL
BBBBBBBBBBBB		AAAAAAA		SSSSSSSSSS		RRRRRRRRRR		TTTTTTTTTTTT		LLL
BBBBBBBBBBBB		AAAAAAA		SSSSSSSSSS		RRRRRRRRRR		TTTTTTTTTTTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRRRRRRRRR		TTT		LLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRRRRRRRRR		TTT		LLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRRRRRRRRR		TTT		LLL
BBB	BBB	AAAAAAAAAAAA			SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAAAAAAAAAAA			SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAAAAAAAAAAA			SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA		SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA		SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA		SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA		SSS	RRR	RRR	TTT		LLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRR	RRR	TTT		LLLLLLLLLLLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRR	RRR	TTT		LLLLLLLLLLLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRR	RRR	TTT		LLLLLLLLLLLL

```
BBBBBBBBB      AAAAAA      SSSSSSSS      CCCCCCCC      TTTTTTTTTT      RRRRRRRR      LL      CCCCCCCC
BBBBBBBBB      AAAAAA      SSSSSSSS      CCCCCCCC      TTTTTTTTTT      RRRRRRRR      LL      CCCCCCCC
BB          BB  AA          AA  SS          CC          TT          RR          RR  LL          CC
BB          BB  AA          AA  SS          CC          TT          RR          RR  LL          CC
BB          BB  AA          AA  SS          CC          TT          RR          RR  LL          CC
BBBBBBBBB      AA          AA  SSSSSS      CC          TT          RRRRRRRR      LL          CC
BBBBBBBBB      AA          AA  SSSSSS      CC          TT          RRRRRRRR      LL          CC
BB          BB  AAAAAAAAAA      SS          CC          TT          RR  RR      LL          CC
BB          BB  AAAAAAAAAA      SS          CC          TT          RR  RR      LL          CC
BB          BB  AA          AA  SS          CC          TT          RR  RR      LL          CC
BB          BB  AA          AA  SS          CC          TT          RR  RR      LL          CC
BBBBBBBBB      AA          AA  SSSSSSSS      CCCCCCCC      TT          RR          RR  LLLLLLLLLL      CCCCCCCC
BBBBBBBBB      AA          AA  SSSSSSSS      CCCCCCCC      TT          RR          RR  LLLLLLLLLL      CCCCCCCC
```

```
LL          IIIIII      SSSSSSSS
LL          IIIIII      SSSSSSSS
LL          II          SS
LL          II          SS
LL          II          SS
LL          II          SS
LL          II          SSSSSS
LL          II          SSSSSS
LL          II          SS
LL          II          SS
LL          II          SS
LL          II          SS
LLLLLLLLLL  IIIIII      SSSSSSSS
LLLLLLLLLL  IIIIII      SSSSSSSS
```



```

1 0001 0 MODULE BAS$CTRLC (
2 0002 0 IDENT = '2-005'
3 0003 0 ) =
4 0004 1 BEGIN
5 0005 1
6 0006 1 *****
7 0007 1 *
8 0008 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
9 0009 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
10 0010 1 * ALL RIGHTS RESERVED.
11 0011 1 *
12 0012 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
13 0013 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
14 0014 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
15 0015 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
16 0016 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
17 0017 1 * TRANSFERRED.
18 0018 1 *
19 0019 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
20 0020 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
21 0021 1 * CORPORATION.
22 0022 1 *
23 0023 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
24 0024 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
25 0025 1 *
26 0026 1 *
27 0027 1 *****
28 0028 1
29 0029 1
30 0030 1 ++
31 0031 1 FACILITY: VAX-11 BASIC Miscellaneous Support
32 0032 1
33 0033 1 ABSTRACT:
34 0034 1
35 0035 1 This module contains routines for enabling, disabling, and
36 0036 1 handling Control C interrupts.
37 0037 1
38 0038 1 ENVIRONMENT: VAX-11 User Mode
39 0039 1
40 0040 1 AUTHOR: John Sauter, CREATION DATE: 19-FEB-1979
41 0041 1
42 0042 1 MODIFIED BY:
43 0043 1
44 0044 1 1-001 - Original. JBS 19-FEB-1979
45 0045 1 1-002 - Add a handler to the AST routine to catch UNWINDS, making
46 0046 1 sure that they dismiss the AST properly. JBS 20-FEB-1979
47 0047 1 1-003 - Add BAS$CTRLC_INIT, for the RUN command. JBS 22-JUN-1979
48 0048 1 1-004 - If a control C trap goes off but the user was not enabled,
49 0049 1 signal an INFO message to the keyboard monitor, who may
50 0050 1 wish to continue. JBS 14-SEP-1979
51 0051 1 1-005 - Use SYS$INPUT rather than TI. JBS 20-SEP-1979
52 0052 1 1-006 - Call SYS$CLRAST to clear the AST, rather than using CHMK.
53 0053 1 JBS 27-NOV-1979
54 0054 1 1-007 - Do translations of SYS$INPUT until it fails to translate.
55 0055 1 JBS 24-JUL-1980
56 0056 1 1-008 - Clear the AST immediately in CONTROL_C. PLL 7-Aug-81
57 0057 1 1-009 - Use LIB$GET_EF to obtain event flags for $QIOWs. PLL 30-Nov-81

```

```
: 58      0058 1 : 1-010 - Don't turn off control c's when a control c AST goes off.
: 59      0059 1 : They should be turned off only by the RCTRLC function. PLL 22-Jun-82
: 60      0060 1 : 1-011 - Edit 010 should also have checked RUN CMD in CONTROL_C, so that
: 61      0061 1 : ctrlc's are always enabled in immediate mode from the VMS point of
: 62      0062 1 : view. PLL 6-Jul-1982
: 63      0063 1 : 1-012 - make ERN and ERL available when user hits CTRL/C from inside
: 64      0064 1 : the environment. MDL 22-Jul-1982
: 65      0065 1 : 2-001 - rewrite to use permanent AST enabling. Also allow CTRLC function
: 66      0066 1 : to work when program runs from a command procedure. MDL 28-Sep-1983
: 67      0067 1 : 2-002 - don't use SYS$CLRAST - it causes us to never return to where we
: 68      0068 1 : were before the AST occurred. MDL 4-Jan-1984
: 69      0069 1 : 2-003 - check if I/O in progress before signalling at AST level, and simply
: 70      0070 1 : return if so. add new routine BAS$$SIGNAL_CTRL_C for use from REC
: 71      0071 1 : level I/O routines. Coordinated change with BAS$$REC_PROC 1-093.
: 72      0072 1 : MDL 12-Mar-1984
: 73      0073 1 : 2-004 - RMS will only return RMSS$CONTROL_C for an interrupted terminal I/O,
: 74      0074 1 : therefore we must signal in all other cases. MDL 3-Apr-1984
: 75      0075 1 : 2-005 - only signal if we're really enabled. MDL 10-Apr-1984
: 76      0076 1 : --
: 77      0077 1 :
: 78      0078 1 : !<BLF/PAGE>
```



```

80 0079 1 |
81 0080 1 | SWITCHES:
82 0081 1 |
83 0082 1 |
84 0083 1 | SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
85 0084 1 |
86 0085 1 |
87 0086 1 | LINKAGES:
88 0087 1 |
89 0088 1 |     NONE
90 0089 1 |
91 0090 1 | TABLE OF CONTENTS:
92 0091 1 |
93 0092 1 |
94 0093 1 | FORWARD ROUTINE
95 0094 1 |     BAS$CTRLC,           ! Enable Control C interrupts
96 0095 1 |     BAS$RCTRLC,         ! Disable Control C interrupts
97 0096 1 |     BAS$$CTRLC_INIT : NOVALUE, ! Set up for RUN command
98 0097 1 |     BAS$$SIGNAL_CTRL_C : NOVALUE, ! Signal the CTRL/C condition
99 0098 1 |     CONTROL_C : NOVALUE;    ! Handle a Control C interrupt
100 0099 1 |
101 0100 1 |
102 0101 1 | INCLUDE FILES:
103 0102 1 |
104 0103 1 |
105 0104 1 | REQUIRE 'RTLIN:RTLPSECT';      ! Macros for defining psects
106 0199 1 |
107 0200 1 | REQUIRE 'RTLIN:BASFRAME';      ! BASIC frame definitions
108 0403 1 |
109 0404 1 | REQUIRE 'RTLML:OTSLUB';        ! LUB definitions
110 0544 1 |
111 0545 1 | REQUIRE 'RTLIN:OTSLNK';        ! Linkage definitions
112 0974 1 |
113 0975 1 | LIBRARY 'RTLSTARLE';          ! Define system symbols
114 0976 1 |
115 0977 1 |
116 0978 1 | MACROS:
117 0979 1 |
118 0980 1 |     NONE
119 0981 1 |
120 0982 1 | EQUATED SYMBOLS:
121 0983 1 |
122 0984 1 |     NONE
123 0985 1 |
124 0986 1 | PSECTS:
125 0987 1 |
126 0988 1 | DECLARE_PSECTS (BAS);          ! Declare psects for BAS$ facility
127 0989 1 |
128 0990 1 | OWN STORAGE:
129 0991 1 |
130 0992 1 |
131 0993 1 | OWN
132 0994 1 |     TT_CHAN : UNSIGNED WORD INITIAL (WORD (0)), ! The channel the terminal is assigned on
133 0995 1 |     RUN_CMD : BYTE INITIAL (BYTE (0)),         ! Set if we are in the RUN command
134 0996 1 |     CC_REALLY_ENABLED : BYTE INITIAL (BYTE (0)), ! Set if the user has control C traps enabled
135 0997 1 |     CC_ENABLED_USER_PT_OF_VIEW : BYTE INITIAL (BYTE (0)); ! Set if the user thinks he has ctrl/c enabled
136 0998 1 |
```

```
137 0999 1
138 1000 1
139 1001 1 ! EXTERNAL REFERENCES:
140 1002 1 !
141 1003 1
142 1004 1 EXTERNAL ROUTINE
143 1005 1     LIB$GET_EF,           ! allocate an event flag
144 1006 1     LIB$FREE_EF,       ! deallocate an event flag
145 1007 1     LIB$SIGNAL,        ! Signal a condition
146 1008 1     LIB$STOP : NOVALUE, ! Signal a fatal error
147 1009 1     LIB$MATCH_COND,    ! Match condition codes
148 1010 1     BAS$$CB_PUSH : JSB_CB_PUSH NOVALUE, ! Load register CCB
149 1011 1     BAS$$CB_POP : JSB_CB_POP NOVALUE,   ! Release register CCB
150 1012 1     BAS$$LINE,         ! get current line
151 1013 1     BAS$$MODULE,       ! get current module name
152 1014 1     BAS$HANDLER;       ! just need address of this
153 1015 1
154 1016 1 EXTERNAL
155 1017 1     BAS$T_ERN : BLOCK [8, BYTE] ,        ! descriptor for module name
156 1018 1     BAS$L_ERR ,                       ! current error code
157 1019 1     BAS$L_ERL ,                       ! line number of error
158 1020 1     OT$$$V_IOINPROG : VOLATILE BITVECTOR; ! channels w/ I/O in progress
159 1021 1
160 1022 1
161 1023 1 !+
162 1024 1 ! The following are the error codes used in this module.
163 1025 1 !-
164 1026 1
165 1027 1 EXTERNAL LITERAL
166 1028 1     BAS$K_PROC__TRA,
167 1029 1     BAS$ _PROC__TRA;
168 1030 1 ! Programmable ^C trap
```



```
170 1031 1 GLOBAL ROUTINE BAS$CTRLC =
171 1032 1
172 1033 1 ! Enable Control C interrupts
173 1034 1
174 1035 1 ++
175 1036 1 FUNCTIONAL DESCRIPTION:
176 1037 1 Enable Control C traps, so that a Control C will cause the
177 1038 1 user's program to take an ON ERROR GOTO branch.
178 1039 1
179 1040 1 FORMAL PARAMETERS:
180 1041 1 NONE
181 1042 1
182 1043 1 IMPLICIT INPUTS:
183 1044 1
184 1045 1 NONE
185 1046 1
186 1047 1 IMPLICIT OUTPUTS:
187 1048 1
188 1049 1 NONE
189 1050 1
190 1051 1 ROUTINE VALUE:
191 1052 1
192 1053 1 Always returns zero.
193 1054 1
194 1055 1 SIDE EFFECTS:
195 1056 1
196 1057 1 Leaves Control C traps enabled if the process has a terminal.
197 1058 1
198 1059 1 --
199 1060 1
200 1061 2 BEGIN
201 1062 2
202 1063 2 +
203 1064 2 If CTRL/C reception is not currently enabled, begin some investigation.
204 1065 2 -
205 1066 2 IF ( NOT .CC_REALLY_ENABLED )
206 1067 2 THEN
207 1068 2 BEGIN
208 1069 2
209 1070 2 LOCAL
210 1071 2 ASSIGN_RESULT,
211 1072 2 QIO_RESULT,
212 1073 2 GETDVI_RESULT,
213 1074 2 GETJPI_RESULT,
214 1075 2 STATUS,
215 1076 2
216 1077 2 EVENT_FLAG,
217 1078 2
218 1079 2 CONTROL_CHARS : VECTOR [2, LONG] INITIAL (0, 8),
219 1080 2
220 1081 2 DEVICE_CLASS : INITIAL(0),
221 1082 2 DEVNAM_DESC : BLOCK [8, BYTE],
222 1083 2 DVI_RETURN_LENGTH : INITIAL(0),
223 1084 2 DVI_ITEMS : VECTOR [4, LONG] INITIAL ( ((DVI$DEVCLASS*16) OR 4),
224 1085 2 DEVICE_CLASS,
225 1086 2 DVI_RETURN_LENGTH,
226 1087 2 0 );
```

```
227 1088 3
228 1089 3
229 1090 3
230 1091 3
231 1092 3
232 1093 3
233 1094 3
234 1095 3
235 1096 3
236 1097 3
237 1098 3
238 1099 3
239 1100 3
240 1101 3
241 1102 3
242 1103 3
243 1104 3
244 1105 3
245 1106 3
246 1107 3
247 P 1108 3
248 P 1109 3
249 P 1110 3
250 1111 3
251 1112 4
252 1113 3
253 1114 3
254 1115 3
255 1116 3
256 1117 3
257 1118 3
258 1119 3
259 1120 3
260 1121 3
261 1122 3
262 1123 3
263 1124 4
264 1125 4
265 1126 4
266 1127 4
267 1128 4
268 1129 4
269 1130 5
270 1131 5
271 1132 5
272 1133 6
273 1134 5
274 1135 4
275 1136 4
276 1137 4
277 1138 4
278 1139 4
279 1140 4
280 1141 4
281 1142 4
282 P 1143 4
283 P 1144 4

      TERMINAL_NAME : VECTOR [256, BYTE],
      JPI_RETURN_LENGTH : INITIAL(0),
      JPI_ITEMS : VECTOR [4, LONG] INITIAL ( ((JPI$ TERMINAL^16) OR 256),
      TERMINAL_NAME,
      JPI_RETURN_LENGTH,
      0 );

+ see if SYSS$INPUT is a terminal device.
-
      DEVNAM_DESC [DSC$W_LENGTH] = %CHARCOUNT ('SYSS$INPUT');
      DEVNAM_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
      DEVNAM_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
      DEVNAM_DESC [DSC$A_POINTER] = TERMINAL_NAME [0];
      CH$MOVE (%CHARCOUNT ('SYSS$INPUT'), CH$PTR (UPLIT ('SYSS$INPUT')), TERMINAL_NAME [0]);

      STATUS = LIB$GET_EF (EVENT_FLAG);
      IF (NOT .STATUS) THEN LIB$STOP (.STATUS);

      GETDVI_RESULT = $GETDVI (EFN = .EVENT_FLAG,
      DEVNAM = DEVNAM_DESC,
      ITMLST = DVI_ITEMS );

      IF ( (NOT .GETDVI_RESULT) OR .DVI_RETURN_LENGTH EQL 0 )
      THEN LIB$STOP (.GETDVI_RESULT);

      STATUS = LIB$FREE_EF (EVENT_FLAG);
      IF (NOT .STATUS) THEN LIB$STOP (.STATUS);

+ If SYSS$INPUT is indeed a terminal device, go ahead and enable CTRL/C
trapping to it.
-
      IF .DEVICE_CLASS EQL DC$_TERM
      THEN
      BEGIN
+ assign a channel to the terminal, if one doesn't already exist.
-
      IF .TT_CHAN EQL 0
      THEN
      BEGIN
      ASSIGN_RESULT = $ASSIGN (DEVNAM = DEVNAM_DESC, CHAN = TT_CHAN);

      IF ( NOT .ASSIGN_RESULT)
      THEN LIB$STOP (.ASSIGN_RESULT);
      END;

+ issue the QIO enabling CTRL/C reception.
-
      STATUS = LIB$GET_EF (EVENT_FLAG);
      IF (NOT .STATUS) THEN LIB$STOP (.STATUS);

      QIO_RESULT = $QIOW (EFN = .EVENT_FLAG,
      CHAN = .TT_CHAN,
```



```
284 P 1145 4 FUNC = (IOS$ SETMODE OR IOSM_OUTBAND OR IOSM_TT_ABORT),
285 P 1146 4 P1 = CONTROL_C,
286 1147 4 P2 = CONTROL_CHARS);
287 1148 4
288 1149 5 IF ( NOT .QIO_RESULT)
289 1150 4 THEN LIB$STOP (.QIO_RESULT);
290 1151 4
291 1152 4 STATUS = LIB$FREE_EF (EVENT_FLAG);
292 1153 4 IF (NOT .STATUS) THEN LIB$STOP (.STATUS);
293 1154 4
294 1155 4 !+
295 1156 4 !- indicate CTRL/C reception is now enabled.
296 1157 4 !-
297 1158 4 CC REALLY_ENABLED = 1;
298 1159 4 END
299 1160 4
300 1161 3 ELSE
301 1162 4 BEGIN
302 1163 4 !+
303 1164 4 !- otherwise, see if the process owns a terminal at all.
304 1165 4 !-
305 1166 4 STATUS = LIB$GET_EF (EVENT_FLAG);
306 1167 4 IF (NOT .STATUS) THEN LIB$STOP (.STATUS);
307 1168 4
308 P 1169 4 GETJPI_RESULT = $GETJPI (EFN = .EVENT_FLAG,
309 1170 4 ITMLST = JPI_ITEMS );
310 1171 4
311 1172 5 IF (NOT .GETJPI_RESULT)
312 1173 4 THEN LIB$STOP (.GETJPI_RESULT);
313 1174 4
314 1175 4 STATUS = LIB$FREE_EF (EVENT_FLAG);
315 1176 4 IF (NOT .STATUS) THEN LIB$STOP (.STATUS);
316 1177 4
317 1178 4 !+
318 1179 4 !- if so, enable CTRL/C reception to that terminal. Otherwise, we cannot
319 1180 4 !- enable CTRL/C reception.
320 1181 4 !-
321 1182 4 IF .JPI_RETURN_LENGTH NEQ 0
322 1183 4 THEN
323 1184 5 BEGIN
324 1185 5 DEVNAM_DESC [DSC$W_LENGTH] = CH$FIND_CH ( 256,
325 1186 5 CH$PTR (TERMINAL_NAME),
326 1187 5 '' ) -
327 1188 5 CH$PTR (TERMINAL_NAME);
328 1189 5 DEVNAM_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
329 1190 5 DEVNAM_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
330 1191 5 DEVNAM_DESC [DSC$A_POINTER] = TERMINAL_NAME [0];
331 1192 5
332 1193 5 !+
333 1194 5 !- assign a channel to the terminal, if one doesn't already exist.
334 1195 5 !-
335 1196 5 IF .TT_CHAN EQLU 0
336 1197 5 THEN
337 1198 6 BEGIN
338 1199 6 ASSIGN_RESULT = $ASSIGN (DEVNAM = DEVNAM_DESC, CHAN = TT_CHAN);
339 1200 6
340 1201 7 IF ( NOT .ASSIGN_RESULT)
```



```

341      1202 6      THEN LIB$STOP (.ASSIGN_RESULT);
342      1203 5      END;
343      1204 5
344      1205 5      !+ issue the QIO enabling CTRL/C reception.
345      1206 5      !-
346      1207 5
347      1208 5      STATUS = LIB$GET_EF (EVENT_FLAG);
348      1209 5      IF (NOT .STATUS) THEN LIB$STOP (.STATUS);
349      1210 5
350      1211 5      QIO_RESULT = $QIOW (EFN = .EVENT_FLAG,
351      1212 5      CHAN = .TT_CHAN,
352      1213 5      FUNC = (IO$ SETMODE OR IO$M_OUTBAND OR IO$M_TT_ABORT),
353      1214 5      P1 = CONTROL_C,
354      1215 5      P2 = CONTROL_CHARS);
355      1216 5
356      1217 6      IF ( NOT .QIO_RESULT)
357      1218 5      THEN LIB$STOP (.QIO_RESULT);
358      1219 5
359      1220 5      STATUS = LIB$FREE_EF (EVENT_FLAG);
360      1221 5      IF (NOT .STATUS) THEN LIB$STOP (.STATUS);
361      1222 5
362      1223 5      !+ indicate CTRL/C reception is now enabled.
363      1224 5      !-
364      1225 5
365      1226 5      CC_REALLY_ENABLED = 1;
366      1227 4      END;
367      1228 4
368      1229 3      END;      ! Else
369      1230 3
370      1231 2      END;      ! If not CC_REALLY_ENABLED
371      1232 2
372      1233 2      !+ indicate the CTRL/C reception is now enabled from the point of view
373      1234 2      of the user.
374      1235 2      !-
375      1236 2
376      1237 2      CC_ENABLED_USER_PT_OF_VIEW = 1;
377      1238 2
378      1239 2      !+ the CTRLC function always returns zero.
379      1240 2      !-
380      1241 2
381      1242 2      RETURN (0);
382      1243 1      END;

```

! end of BAS\$CTRLC

.TITLE BAS\$CTRLC
.IDENT \2-005\

.PSECT _BAS\$DATA,NOEXE, PIC,2

```

0000 00000 TT_CHAN: .WORD 0
00 00002 RUN_CMD: .BYTE 0
00 00003 CC_REALLY_ENABLED:
      .BYTE 0
00 00004 CC_ENABLED_USER_PT_OF_VIEW:
      .BYTE 0

```

.PSECT _BAS\$CODE,NOWRT, SHR, PIC,2


```
00040004 00000 P.AAA: .LONG 262148
00000000 00004 .LONG 0
00000000 00008 .LONG 0, 0
031D0100 00010 P.AAB: .LONG 52232448
00000000 00014 .LONG 0
00000000 00018 .LONG 0, 0
00 00 00 54 55 50 4E 49 24 53 59 53 00020 P.AAC: .ASCII \SYSS$INPUT\<0><0><0>

.EXTRN LIB$GET_EF, LIB$FREE_EF
.EXTRN LIB$SIGNAL, LIB$STOP
.EXTRN LIB$MATCH_COND, BAS$$CB_PUSH
.EXTRN BAS$$CB_POP, BAS$$LINE
.EXTRN BAS$$MODULE, BAS$HANDLER
.EXTRN BAS$T_ERN, BAS$L_ERR
.EXTRN BAS$L_ERL, OTS$$V_IOINPROG
.EXTRN BAS$K_PROC_TRA
.EXTRN BAS$ PROC_TRA, SYSS$GETDVI
.EXTRN SYSS$ASSIGN, SYSS$QIOW
.EXTRN SYSS$GETJPI

OFFC 00000
5B 00000000G 00 9E 00002 MOVAB SYSS$QIOW, R11
5A 00000000G 00 9E 00009 MOVAB SYSS$ASSIGN, R10
59 00000000G 00 9E 00010 MOVAB LIB$FREE_EF, R9
58 00000000G 00 9E 00017 MOVAB LIB$GET_EF, R8
57 00000000' EF 9E 0001E MOVAB TT_CHAN, R7
56 00000000G 00 9E 00025 MOVAB LIB$STOP, R6
5E FE0 CE 9E 0002C MOVAB -320(SP), SP
03 03 A7 E9 00031 BLBC CC REALLY_ENABLED, 1$
01BB 31 00035 BRW 21$
FC AD F8 AD D4 00038 1$: CLRL CONTROL_CHARS
6E 7C 0003F MOVL #8, CONTROL_CHARS+4
E0 AD 8F AF 10 28 00041 CLRQ DEVICE_CLASS
E4 AD 6E 9E 00047 MOVAB #16, P.AAA, DVI_ITEMS
E8 AD 04 AE 9E 0004B MOVAB DEVICE_CLASS, DVI_ITEMS+4
10 AE 8D AF 10 28 00050 MOVAB DVI_RETURN_LENGTH, DVI_ITEMS+8
14 AE 20 AE 9E 00059 CLRL JPI_RETURN_LENGTH
18 AE 08 AE 9E 0005E MOVAB #16, P.AAB, JPI_ITEMS
F0 AD 010E0009 8F D0 00063 MOVAB TERMINAL_NAME, JPI_ITEMS+4
F4 AD 20 AE 9E 00068 MOVAB JPI_RETURN_LENGTH, JPI_ITEMS+8
20 AE 80 AF 09 28 00070 MOVL #17694729, DEVNAM_DESC
OC AE 9F 00076 MOVAB TERMINAL_NAME, DEVNAM_DESC+4
68 01 FB 00079 MOVAB #9, P.AAC, TERMINAL_NAME
52 50 D0 0007C PUSHAB EVENT_FLAG
05 52 E8 0007F CALLS #1, LIB$GET_EF
66 52 DD 00082 MOVL R0, STATUS
01 FB 00084 BLBS STATUS, 2$
7E 7C 00087 2$: CALLS #1, LIB$STOP
7E 7C 00089 CLRL -(SP)
E0 AD 9F 0008B CLRL -(SP)
F0 AD 9F 0008E PUSHAB DVI_ITEMS
28 AE DD 00091 PUSHAB DEVNAM_DESC
7E D4 00091 CLRL -(SP)
28 AE DD 00093 PUSHL EVENT_FLAG
```


00000000G	00	08	FB	00096	CALLS	#8, SYSS\$GETDVI	:	1112	
	05	50	E9	0009D	BLBC	GETDVI_RESULT, 3\$:		
		04	AE	D5	000A0	TSTL	DVI_RETURN_LENGTH	:	
		05	12	000A3	BNEQ	4\$:		
	66	50	DD	000A5	3\$: PUSHL	GETDVI_RESULT	:	1113	
		01	FB	000A7	CALLS	#1, LIB\$STOP	:		
	69	0C	AE	9F	000AA	4\$: PUSHAB	:	1115	
	52	01	FB	000AD	CALLS	#1, LIB\$FREE_EF	:		
	05	50	D0	000B0	MOVL	R0, STATUS	:		
		52	E8	000B3	BLBS	STATUS, 5\$:	1116	
		52	DD	000B6	PUSHL	STATUS	:		
	66	01	FB	000B8	CALLS	#1, LIB\$STOP	:		
00000042	8F	6E	D1	000BB	5\$: CMPL	DEVICE_CLASS, #66	:	1122	
		64	12	000C2	BNEQ	10\$:		
		67	B5	000C4	TSTW	TT_CHAN	:	1128	
		15	12	000C6	BNEQ	6\$:		
		7E	7C	000C8	CLRQ	-(SP)	:	1131	
		57	DD	000CA	PUSHL	R7	:		
		F0	AD	9F	000CC	PUSHAB	DEVNAM_DESC	:	
	6A	04	FB	000CF	CALLS	#4, SYSS\$ASSIGN	:		
	54	50	D0	000D2	MOVL	R0, ASSIGN_RESULT	:		
	05	54	E8	000D5	BLBS	ASSIGN_RESULT, 6\$:	1133	
		54	DD	000D8	PUSHL	ASSIGN_RESULT	:	1134	
	66	01	FB	000DA	CALLS	#1, LIB\$STOP	:		
		0C	AE	9F	000DD	6\$: PUSHAB	:	1140	
	68	01	FB	000E0	CALLS	#1, LIB\$GET_EF	:		
	52	50	D0	000E3	MOVL	R0, STATUS	:		
	05	52	E8	000E6	BLBS	STATUS, 7\$:	1141	
		52	DD	000E9	PUSHL	STATUS	:		
	66	01	FB	000EB	CALLS	#1, LIB\$STOP	:		
		7E	7C	000EE	7\$: CLRQ	-(SP)	:	1147	
		7E	7C	000F0	CLRQ	-(SP)	:		
		F8	AD	9F	000F2	PUSHAB	CONTROL_CHARS	:	
		0000V	CF	9F	000F5	PUSHAB	CONTROL_C	:	
			7E	7C	000F9	CLRQ	-(SP)	:	
			7E	D4	000FB	CLRL	-(SP)	:	
	7E	1423	8F	3C	000FD	MOVZWL	#5155, -(SP)	:	
	7E		67	3C	00102	MOVZWL	TT_CHAN, -(SP)	:	
		38	AE	DD	00105	PUSHL	EVENT_FLAG	:	
	6B		0C	FB	00108	CALLS	#12, SYSS\$QIOW	:	
	53		50	D0	0010B	MOVL	R0, QIO_RESULT	:	
	05		53	E8	0010E	BLBS	QIO_RESULT, 8\$:	1149
			53	DD	00111	PUSHL	QIO_RESULT	:	1150
	66		01	FB	00113	CALLS	#1, LIB\$STOP	:	
		0C	AE	9F	00116	8\$: PUSHAB	:	1152	
	69		01	FB	00119	CALLS	#1, LIB\$FREE_EF	:	
	52		50	D0	0011C	MOVL	R0, STATUS	:	
	03		52	E9	0011F	BLBC	STATUS, 9\$:	1153
			00CA	31	00122	BRW	20\$:	
			00C2	31	00125	BRW	19\$:	
		0C	AE	9F	00128	10\$: PUSHAB	:	1166	
	68		01	FB	0012B	CALLS	#1, LIB\$GET_EF	:	
	52		50	D0	0012E	MOVL	R0, STATUS	:	
	05		52	E8	00131	BLBS	STATUS, 11\$:	1167
			52	DD	00134	PUSHL	STATUS	:	
	66		01	FB	00136	CALLS	#1, LIB\$STOP	:	
			7E	7C	00139	11\$: CLRQ	-(SP)	:	1170

				7E	D4	0013B	CLRL	-(SP)		
			1C	AE	9F	0013D	PUSHAB	JPI_ITEMS		
				7E	7C	00140	CLRQ	-(SP)		
			24	AE	DD	00142	PUSHL	EVENT_FLAG		
	00000000G	00		07	FB	00145	CALLS	#7, SYSS\$GETJPI		
		05		50	E8	0014C	BLBS	GETJPI_RESULT, 12\$		1172
				50	DD	0014F	PUSHL	GETJPI_RESULT		1173
		66		01	FB	00151	CALLS	#1, LIB\$STOP		
			0C	AE	9F	00154	PUSHAB	EVENT_FLAG		1175
		69		01	FB	00157	CALLS	#1, LIB\$FREE_EF		
		52		50	D0	0015A	MOVL	R0, STATUS		
		05		52	E8	0015D	BLBS	STATUS, 13\$		1176
				52	DD	00160	PUSHL	STATUS		
		66		01	FB	00162	CALLS	#1, LIB\$STOP		
			08	AE	D5	00165	TSTL	JPI_RETURN_LENGTH		1182
				03	12	00168	BNEQ	14\$		
				0086	31	0016A	BRW	21\$		
				00	3A	0016D	LOCC	#0, #256, TERMINAL_NAME		1185
				02	12	00174	BNEQ	15\$		
				51	D4	00176	CLRL	R1		
20	AE	0100	8F				MOVAB	TERMINAL_NAME, R0		1188
				50	AE	9E	SUBW3	R0, R1, DEVNAM_DESC		
FO	AD			51	50	A3	MOVW	#270, DEVNAM_DESC+2		1189
		F2	AD	010E	8F	B0	MOVAB	TERMINAL_NAME, DEVNAM_DESC+4		1191
		F4	AD	20	AE	9E	TSTW	TT_CHAN		1196
				67	B5	0018C	BNEQ	16\$		
				15	12	0018E	CLRQ	-(SP)		1199
				7E	7C	00190	PUSHL	R7		
				57	DD	00192	PUSHAB	DEVNAM_DESC		
			FO	AD	9F	00194	CALLS	#4, SYSS\$ASSIGN		
				04	FB	00197	MOVL	R0, ASSIGN_RESULT		
		6A		50	D0	0019A	BLBS	ASSIGN_RESULT, 16\$		1201
		54		54	E8	0019D	PUSHL	ASSIGN_RESULT		1202
		05		54	DD	001A0	CALLS	#1, LIB\$STOP		
		66		01	FB	001A2	PUSHAB	EVENT_FLAG		1208
			0C	AE	9F	001A5	CALLS	#1, LIB\$GET_EF		
		68		01	FB	001A8	MOVL	R0, STATUS		
		52		50	D0	001AB	BLBS	STATUS, 17\$		1209
		05		52	E8	001AE	PUSHL	STATUS		
				52	DD	001B1	CALLS	#1, LIB\$STOP		
		66		01	FB	001B3	CLRQ	-(SP)		1215
				7E	7C	001B6	CLRQ	-(SP)		
				7E	7C	001B8	PUSHAB	CONTROL_CHARS		
			F8	AD	9F	001BA	PUSHAB	CONTROL_C		
			0000V	CF	9F	001BD	CLRQ	-(SP)		
				7E	7C	001C1	CLRL	-(SP)		
				7E	D4	001C3	MOVZWL	#5155, -(SP)		
		7E		8F	3C	001C5	MOVZWL	TT_CHAN, -(SP)		
		7E		67	3C	001CA	PUSHL	EVENT_FLAG		
			38	AE	DD	001CD	CALLS	#12, SYSS\$QIOW		
		6B		0C	FB	001D0	MOVL	R0, QIO_RESULT		
		53		50	D0	001D3	BLBS	QIO_RESULT, 18\$		1217
		05		53	E8	001D6	PUSHL	QIO_RESULT		1218
				53	DD	001D9	CALLS	#1, LIB\$STOP		
		66		01	FB	001DB	PUSHAB	EVENT_FLAG		1220
			0C	AE	9F	001DE	CALLS	#1, LIB\$FREE_EF		
		69		01	FB	001E1	MOVL	R0, STATUS		
		52		50	D0	001E4				

BAS\$CTRLC
2-005

D 11
16-Sep-1984 00:09:26
14-Sep-1984 11:54:48

VAX-11 Bliss-32 V4.0-742
[BASRTL.SRC]BASCTRLC.B32;1

Page 12
(3)

	05	52	E8	001E7	BLBS	STATUS, 20\$: 1221
		52	DD	001EA	PUSHL	STATUS	: :
	66	01	FB	001EC	CALLS	#1, LIB\$STOP	: :
03	A7	01	90	001EF	MOVB	#1, CC_REALLY_ENABLED	: 1226
04	A7	01	90	001F3	MOVB	#1, CC_ENABLED_USER_PT_OF_VIEW	: 1237
		50	D4	001F7	CLRL	RO	: 1242
		04	001F9	RET			: 1243

; Routine Size: 506 bytes, Routine Base: _BAS\$CODE + 002C

; 383 1244 1


```

: 385      1245 1 GLOBAL ROUTINE BAS$RCTRLC =                ! Disable Control C interrupts
: 386      1246 1
: 387      1247 1 !++
: 388      1248 1 FUNCTIONAL DESCRIPTION:
: 389      1249 1
: 390      1250 1         Disable Control C traps, so that a Control C will cause the
: 391      1251 1         user's program to stop, as usual.
: 392      1252 1
: 393      1253 1 FORMAL PARAMETERS:
: 394      1254 1
: 395      1255 1         NONE
: 396      1256 1
: 397      1257 1 IMPLICIT INPUTS:
: 398      1258 1
: 399      1259 1         NONE
: 400      1260 1
: 401      1261 1 IMPLICIT OUTPUTS:
: 402      1262 1
: 403      1263 1         NONE
: 404      1264 1
: 405      1265 1 ROUTINE VALUE:
: 406      1266 1 COMPLETION CODES:
: 407      1267 1
: 408      1268 1         Always returns zero.
: 409      1269 1
: 410      1270 1 SIDE EFFECTS:
: 411      1271 1
: 412      1272 1         Leaves Control C traps disabled.
: 413      1273 1
: 414      1274 1 --
: 415      1275 1
: 416      1276 2 BEGIN
: 417      1277 2
: 418      1278 2 LOCAL
: 419      1279 2     EVENT_FLAG,
: 420      1280 2     STATUS,
: 421      1281 2     QIO_RESULT;
: 422      1282 2
: 423      1283 2 !+
: 424      1284 2 Only turn CTRL/C reception off if it is currently on, and we're NOT in
: 425      1285 2 the environment (RUN_CMD). CTRL/C reception should always be enabled
: 426      1286 2 (from the point of view of the user) when running in the environment.
: 427      1287 2 -
: 428      1288 2
: 429      1289 3 IF ((.TT_CHAN NEQU 0) AND ( .CC_REALLY_ENABLED ))
: 430      1290 3 THEN
: 431      1291 3 BEGIN
: 432      1292 3 !+
: 433      1293 3 If we are in the RUN command (where control Cs should always remain
: 434      1294 3 enabled) or if control Cs are not enabled, don't issue the QIO.
: 435      1295 3 -
: 436      1296 3
: 437      1297 4 IF ( NOT .RUN_CMD)
: 438      1298 3 THEN
: 439      1299 4 BEGIN
: 440      1300 4
: 441      1301 4     STATUS = LIB$GET_EF (EVENT_FLAG);
```

```
! end of BASSRCTRLC
```

Address	Hex	Op	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op417	Op418
---------	-----	----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------

BAS\$CTRLC
2-005

G 11
16-Sep-1984 00:09:26
14-Sep-1984 11:54:48

VAX-11 Bliss-32 V4.0-742
[BASRTL.SRC]BAS\$CTRLC.B32;1

Page 15
(4)

04 00061

RET

; 1325

; Routine Size: 98 bytes, Routine Base: _BAS\$CODE + 0226

; 466 1326 1

```

468 1327 1 GLOBAL ROUTINE BAS$CTRLC_INIT : NOVALUE =      ! Set up for RUN command
469 1328 1
470 1329 1 ++
471 1330 1 FUNCTIONAL DESCRIPTION:
472 1331 1
473 1332 1      Set up for the RUN environment. Since this image is to run under the RUN
474 1333 1      command, control C traps are always enabled, from the point of view of
475 1334 1      VMS. If one goes off when the user has not enabled for control C traps,
476 1335 1      the user is not allowed to intercept the signal (because of its severity)
477 1336 1      and the keyboard monitor gets it instead.
478 1337 1
479 1338 1 FORMAL PARAMETERS:
480 1339 1
481 1340 1      NONE
482 1341 1
483 1342 1 IMPLICIT INPUTS:
484 1343 1
485 1344 1      NONE
486 1345 1
487 1346 1 IMPLICIT OUTPUTS:
488 1347 1
489 1348 1      NONE
490 1349 1
491 1350 1 ROUTINE VALUE:
492 1351 1 COMPLETION CODES:
493 1352 1
494 1353 1      NONE
495 1354 1
496 1355 1 SIDE EFFECTS:
497 1356 1
498 1357 1      Leaves Control C traps disabled from the user's point of view, but
499 1358 1      enabled from VMS's point of view.
500 1359 1
501 1360 1 --
502 1361 1
503 1362 2 BEGIN
504 1363 2
505 1364 2 + Make sure the $ASSIGN and $QIO have been done.
506 1365 2 -
507 1366 2 BAS$CTRLC ();
508 1367 2 +
509 1368 2 Flag that we are in the RUN environment. This will prevent the
510 1369 2 Control C enable from being turned off, from the point of view
511 1370 2 of VMS.
512 1371 2 -
513 1372 2 RUN_CMD = 1;
514 1373 2 +
515 1374 2 Turn control C enable off from the user's point of view.
516 1375 2 -
517 1376 2 BAS$RCTRLC ();
518 1377 2 RETURN;
519 1378 1 END;

```


BAS\$CTRLC
2-005

I 11
16-Sep-1984 00:09:26 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 11:54:48 [BASRTL.SRC]BASCTRLC.B32;1

Page 17
(5)

	FD9D	CF	0000	00000
00000000	8C	EF	00	FB 00002
		AF	01	90 00007
			00	FB 0000E
			04	00012

.ENTRY BAS\$CTRLC INIT, Save nothing
CALLS #0, BAS\$CTRLC
MOVB #1, RUN CMD
CALLS #0, BAS\$RCTRLC
RET

: 1327
: 1366
: 1372
: 1376
: 1378

; Routine Size: 19 bytes, Routine Base: _BAS\$CODE + 0288

; 520 1379 1

```
522 1380 1 GLOBAL ROUTINE BAS$$SIGNAL_CTRLC : NOVALUE = ! Signal CTRL/C
523 1381 1
524 1382 1 !++
525 1383 1 FUNCTIONAL DESCRIPTION:
526 1384 1
527 1385 1 Signals CTRL/C to the BASIC program.
528 1386 1
529 1387 1 FORMAL PARAMETERS:
530 1388 1
531 1389 1 NONE
532 1390 1
533 1391 1 IMPLICIT INPUTS:
534 1392 1
535 1393 1 NONE
536 1394 1
537 1395 1 IMPLICIT OUTPUTS:
538 1396 1
539 1397 1 NONE
540 1398 1
541 1399 1 ROUTINE VALUE:
542 1400 1 COMPLETION CODES:
543 1401 1
544 1402 1 NONE
545 1403 1
546 1404 1 SIDE EFFECTS:
547 1405 1
548 1406 1 Calls the user's code by Signaling.
549 1407 1 If the user is not enabled (which means that the program must
550 1408 1 be being run under the RUN command) then the signal goes to
551 1409 1 the keyboard monitor, which may do a continue or an unwind.
552 1410 1
553 1411 1 --
554 1412 1
555 1413 2 BEGIN
556 1414 2
557 1415 2 LOCAL
558 1416 2 COND_VAL : BLOCK [4, BYTE],
559 1417 2 FMP : REF BLOCK [0, BYTE] FIELD (BSF$FCD),
560 1418 2 MOD_NAME_ADDR;
561 1419 2
562 1420 2 BUILTIN
563 1421 2 FP;
564 1422 2
565 1423 2 !+
566 1424 2 if we're not really enabled, don't bother signalling.
567 1425 2 !-
568 1426 2 IF NOT .CC_REALLY_ENABLED
569 1427 2 THEN
570 1428 2 RETURN;
571 1429 2
572 1430 2 !+
573 1431 2 Search for a BASIC major frame.
574 1432 2 !-
575 1433 2 FMP = .FP;
576 1434 2
577 1435 3 WHILE ( (.FMP NEQ 0) AND (.FMP [BSF$A_HANDLER] NEQA BAS$HANDLER) )
578 1436 2 DO
```



```
579 1437 3 BEGIN
580 1438 FMP = .FMP [BSF$A_SAVED_FP];
581 1439 END;
582 1440
583 1441 +
584 1442 get current error line (ERL) and error module (ERN$), and define current
585 1443 error as 'Programmable ^C trap'.
586 1444 -
587 1445 IF (.FMP NEQ 0)
588 1446 THEN
589 1447 BEGIN
590 1448 BAS$ _ERL = BAS$$LINE (.FMP);
591 1449 MOD_NAME_ADDR = BAS$$MODULE (.FMP);
592 1450 BAS$T_ERN [DSC$A_POINTER] = .MOD_NAME_ADDR + 1;
593 1451 BAS$T_ERN [DSC$W_LENGTH] = .BLOCK [.MOD_NAME_ADDR, 0, 0, 8, 0; 1, BYTE];
594 1452 BAS$T_ERN [DSC$B_CLASS] = DSC$K_CLASS_S;
595 1453 BAS$T_ERN [DSC$B_DTYPE] = DSC$K_DTYPE_T;
596 1454 BAS$ _ERR = BAS$R_PROC__TRA;
597 1455 END;
598 1456
599 1457 +
600 1458 Now signal the appropriate BASIC condition for Control C. By default, the
601 1459 severity for CTRL/C is ERROR. If the user is not enabled, signal information.
602 1460 BAS$HANDLER will gain control when the exception is signalled, and check the
603 1461 severity. If it is ERROR, then the assumption is that the user has a handler
604 1462 for CTRL/C and the user's handler is called. Otherwise (informational),
605 1463 control will be returned to KMON (environment) or DCL (run from DCL).
606 1464 -
607 1465 COND_VAL = BAS$ _PROC__TRA;
608 1466
609 1467 IF ( NOT .CC_ENABLED_USER_PT_OF_VIEW)
610 1468 THEN COND_VAL [ST$S$V_SEVERITY] = ST$S$K_INFO;
611 1469
612 1470 LIB$SIGNAL (.COND_VAL);
613 1471
614 1472 +
615 1473 If we get to here, then the program was being run from the environment, the
616 1474 user had no CTRL/C handler, and the keyboard monitor received the CONTINUE
617 1475 command. Dismiss the AST (done automatically by returning).
618 1476 -
619 1477 RETURN;
620 1478 END;

! end of BAS$$SIGNAL_CTRL C
```

53	00000000G	00	9E	00002	.ENTRY	BAS\$\$SIGNAL_CTRL C, Save R2,R3	: 1380
69	00000000'	EF	E9	00009	MOVAB	BAS\$T_ERN+4, R3	: 1426
52		5D	D0	00010	BLBC	CC REALLY_ENABLED, 5\$: 1433
		12	13	00013	MOVL	FP, FMP	: 1435
50	00000000G	00	9E	00015	BEQL	2\$	
50		62	D1	0001C	MOVAB	BAS\$HANDLER, R0	
		06	13	0001F	CMPL	(FMP), R0	
52	0C	A2	D0	00021	BEQL	2\$	
		EC	11	00025	MOVL	12(FMP), FMP	: 1438
					BRB	1\$: 1435

BAS\$CTRLC
2-005

L 11
16-Sep-1984 00:09:26
14-Sep-1984 11:54:48

VAX-11 Bliss-32 V4.0-742
[BASRTL.SRC]BASCTRLC.B32;1

Page 20
(6)

				52	D5	00027	2\$:	TSTL	FMP		: 1445
				32	13	00029		BEQL	3\$: 1448
				52	DD	0002B		PUSHL	FMP		: 1449
	00000000G	00		01	FB	0002D		CALLS	#1, BAS\$\$LINE		: 1450
	00000000G	00		50	D0	00034		MOVL	R0, BAS\$L_ERL		: 1451
				52	DD	0003B		PUSHL	FMP		: 1453
	00000000G	00		01	FB	0003D		CALLS	#1, BAS\$\$MODULE		: 1454
		63	01	A0	9E	00044		MOVAB	1(R0), BAS\$T_ERN+4		: 1465
	FC	A3		60	9B	00048		MOVZBW	(MOD_NAME_ADDR), BAS\$T_ERN		: 1467
	FE	A3	010E	8F	B0	0004C		MOVW	#270, BAS\$T_ERN+2		: 1470
	00000000G	00	00000000G	8F	D0	00052		MOVL	#BAS\$K_PROC_TRA, BAS\$L_ERR		: 1478
		50	00000000G	8F	D0	0005D	3\$:	MOVL	#BAS\$ PROC_TRA, COND_VAL		
		05	00000000'	EF	E8	00064		BLBS	CC_ENABLED_USER_PT_OF_VIEW, 4\$		
50		03	00	03	F0	0006B		INSV	#3, #0, #3, COND_VAL		
				50	DD	00070	4\$:	PUSHL	COND_VAL		
	00000000G	00		01	FB	00072		CALLS	#1, [IB\$SIGNAL		
				04	00079	5\$:		RET			

; Routine Size: 122 bytes, Routine Base: _BAS\$CODE + 029B

; 621 1479 1


```

: 623      1480 1 ROUTINE CONTROL_C : NOVALUE =                ! Handle a Control C interrupt
: 624      1481 1
: 625      1482 1 ++
: 626      1483 1 FUNCTIONAL DESCRIPTION:
: 627      1484 1
: 628      1485 1      This is the RTL AST routine for CTRL/C's deliered to BASIC programs.
: 629      1486 1      It handles the Control C interrupt, and may signal it to the BASIC
: 630      1487 1      program, depending on whether I/O was interrupted or not.
: 631      1488 1
: 632      1489 1 FORMAL PARAMETERS:
: 633      1490 1
: 634      1491 1      NONE
: 635      1492 1
: 636      1493 1 IMPLICIT INPUTS:
: 637      1494 1
: 638      1495 1      NONE
: 639      1496 1
: 640      1497 1 IMPLICIT OUTPUTS:
: 641      1498 1
: 642      1499 1      NONE
: 643      1500 1
: 644      1501 1 ROUTINE VALUE:
: 645      1502 1 COMPLETION CODES:
: 646      1503 1
: 647      1504 1      NONE
: 648      1505 1
: 649      1506 1 SIDE EFFECTS:
: 650      1507 1
: 651      1508 1      May call the user's code by Signaling.
: 652      1509 1
: 653      1510 1 --
: 654      1511 1
: 655      1512 2 BEGIN
: 656      1513 2
: 657      1514 2 GLOBAL REGISTER
: 658      1515 2      CCB = K_CCB_REG : REF BLOCK [, BYTE];
: 659      1516 2
: 660      1517 2 LOCAL
: 661      1518 2      COND_VAL : BLOCK [4, BYTE],
: 662      1519 2      FMP : REF BLOCK [0, BYTE] FIELD (BSF$FCD),
: 663      1520 2      MOD_NAME_ADDR;
: 664      1521 2
: 665      1522 2 BUILTIN
: 666      1523 2      FP;
: 667      1524 2
: 668      1525 2 ++
: 669      1526 2 search for I/O active; if I/O is active on any channel then assume
: 670      1527 2 this AST interrupted it.
: 671      1528 2 --
: 672      1529 2 INCR LUN FROM 0 TO LUB$K_LUN_MAX DO
: 673      1530 3 BEGIN
: 674      1531 4 IF ( .OTSS$V_IOINPROG [.LUN] NEQU 0 )
: 675      1532 3 THEN
: 676      1533 3 ++
: 677      1534 3 I/O is active. Push the channel and see if this is a
: 678      1535 3 forcible (i.e., terminal) device.
: 679      1536 3 --
```

```
: 680      1537  4      BEGIN
: 681      1538  4      BAS$$CB_PUSH ( .LUN + LUB$K_ILUN_MIN, LUB$K_ILUN_MIN );
: 682      1539  4      IF .CCB-[LUB$V_FORCIBLE]
: 683      1540  4      THEN
: 684      1541  4          +
: 685      1542  4          | this is indeed a terminal device.  pop this channel and
: 686      1543  4          | return.  the record level routines will notice the
: 687      1544  4          | RMSS$CONTROL_C return status and signal.
: 688      1545  4          |
: 689      1546  4          | note that returning dismisses the AST.
: 690      1547  4          |
: 691      1548  5          BEGIN
: 692      1549  5              BAS$$CB_POP ();
: 693      1550  5              RETURN;
: 694      1551  4          END;
: 695      1552  4      +
: 696      1553  4      | not a terminal device on this channel.  pop the channel
: 697      1554  4      | and continue looking.
: 698      1555  4      |
: 699      1556  4      |
: 700      1557  4      BAS$$CB_POP ();
: 701      1558  4
: 702      1559  3      END;
: 703      1560  2      END;
: 704      1561  2
: 705      1562  2      +
: 706      1563  2      | An I/O was not interrupted, or I/O to a device other than a terminal was
: 707      1564  2      | interrupted.  Signal the CTRL_C condition at this time.
: 708      1565  2      |
: 709      1566  2      BAS$$SIGNAL_CTRL_C();
: 710      1567  2
: 711      1568  2      RETURN;
: 712      1569  1      END;
```

! end of CONTROL_C

				081C 00000 CONTROL_C:				
		54	00000000G	00	9E 00002	.WORD	Save R2,R3,R4,R11	: 1480
				53	D4 00009	MOVAB	BAS\$\$CB_POP, R4	: 1529
50	00000000G	00		53	EF 0000B 1\$:	CLRL	LUN	: 1531
				50	D5 00014	EXTZV	LUN, #1, OTS\$\$V_IOINPROG, R0	
				17	13 00016	TSTL	R0	
		52	F8	A3	9E 00018	BEQL	3\$: 1538
		50		08	CE 0001C	MOVAB	-8(LUN), R2	
			00000000G	00	16 0001F	MNEGL	#8, R0	
	03	FE	AB	06	E1 00025	JSB	BAS\$\$CB_PUSH	: 1539
				64	16 0002A	BBC	#6, -2(CCB), 2\$: 1549
				04	0002C	JSB	BAS\$\$CB_POP	: 1548
				64	16 0002D 2\$:	RET		: 1557
	D4		53 00000077	8F	F3 0002F 3\$:	JSB	BAS\$\$CB_POP	: 1557
		FF4A	CF	00	FB 00037	AOBLEQ	#119, LUN, 1\$: 1529
				04	0003C	CALLS	#0, BAS\$\$SIGNAL_CTRL_C	: 1566
						RET		: 1569

; Routine Size: 61 bytes, Routine Base: _BAS\$CODE + 0315


```
: 713      1570  1
: 714      1571  1 END
: 715      1572  0 ELUDOM
```

! end of module BAS\$CTRLC

PSECT SUMMARY

Name	Bytes	Attributes
BAS\$DATA	5	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, PIC, ALIGN(2)
BAS\$CODE	850	NOVEC, NOWRT, RD, EXE, SHR, LCL, REL, CON, PIC, ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	21	0	581	00:01.1

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS\$:BAS\$CTRLC/OBJ=OBJ\$:BAS\$CTRLC MSRC\$:BAS\$CTRLC/UPDATE=(ENH\$:BAS\$CTRLC)

```
: Size:      806 code + 49 data bytes
: Run Time:   00:19.6
: Elapsed Time: 00:43.2
: Lines/CPU Min: 4802
: Lexemes/CPU-Min: 26578
: Memory Used: 220 pages
: Compilation Complete
```


0020

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY